Acadgild

Full-stack Web Development

Acadgild

Full-stack Web Development

Agenda

1. What are Conditional Statements?
2. If Statement
3. If else Statement
4. If else if Statement
5. Switch Statement
6. What are loops?
7. For.in Loop
8. While.Loop
9. Do.while Loop
10. Loop Control Statements
11. Break Statement
12. Continue Statement

What are Conditional Statements?

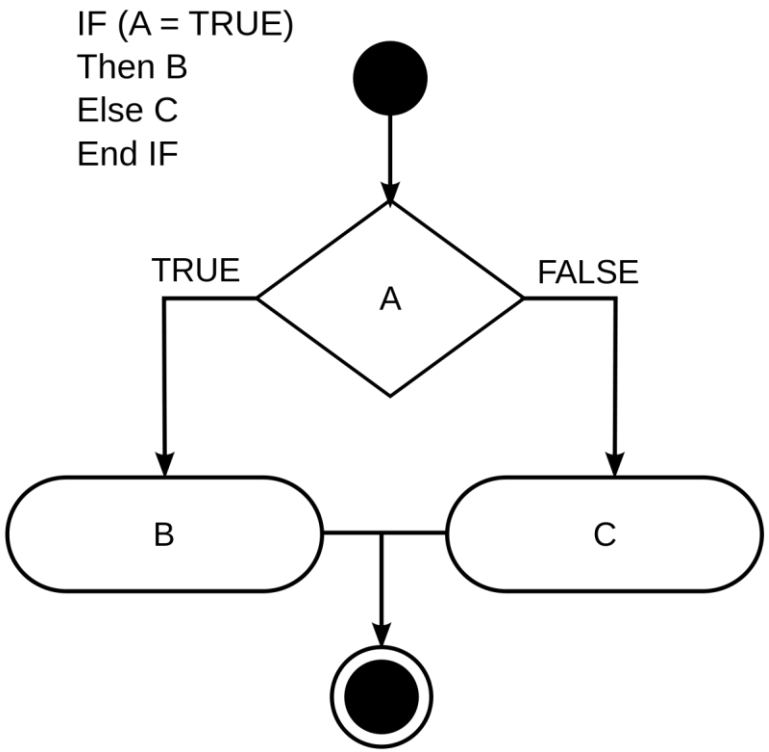
Conditional statements are used to perform different actions based on different conditions.

JavaScript supports the following conditional statements

- **if**
- **if else**
- **if else if**
- **switch**

If Statement

Used to specify a block of JavaScript code to be executed **if a condition is true.**



If Statement

- Syntax

```
if (condition) {
    block of code to be executed if the condition is true
}
```

- Example

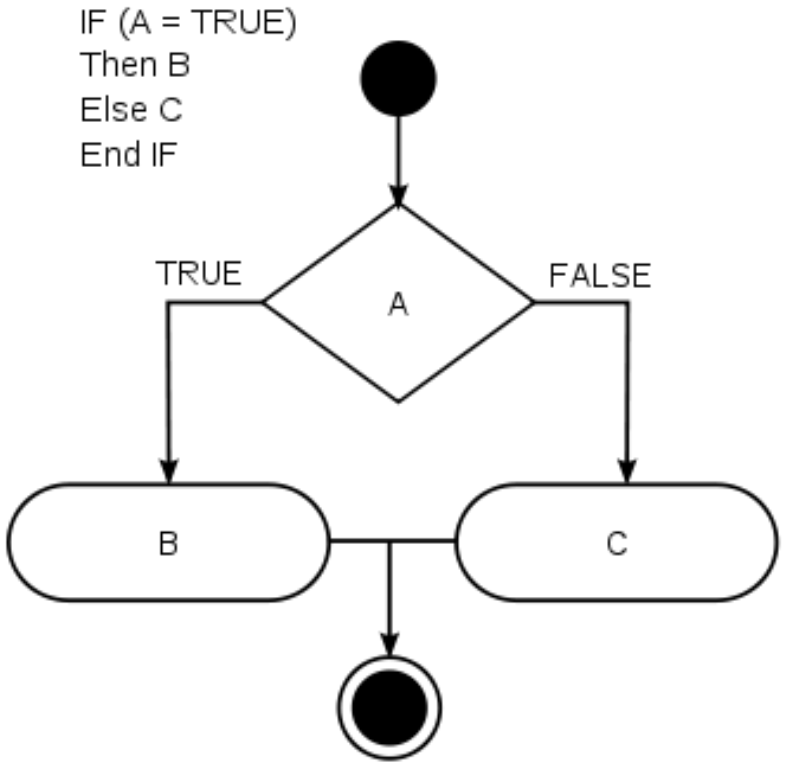
```
<script>
    var age = 12;
    var canVote;
    if (hour < 18) {
        canVote = false;
    }
</script>
```

If else Statement (cont)

Used to specify

one block of JavaScript code to be executed **if a condition is true** and

another block of JavaScript code to be executed **if a condition is false**



If else Statement (cont)

- Syntax

```
if (condition) {
    block of code to be executed if the condition is true
} else {
   block of code to be executed if the condition is false
}
```

- Example

```
<script>
    var age = 12;
    var canVote;
    if (hour < 18) {
        canVote = false;
    } else {
        canVote = true;
    }
</script>
```

If else if Statement

- An advanced form of **if…else**
- Used to specify multiple mutually exclusive conditions
- Syntax

```
if (condition1) {
    block of code to be executed if condition1 is true
} else if (condition2) {
    block of code to be executed if the condition1 is false and condition2 is
true
} else {
    block of code to be executed if the condition1 is false and condition2 is
false
}
```

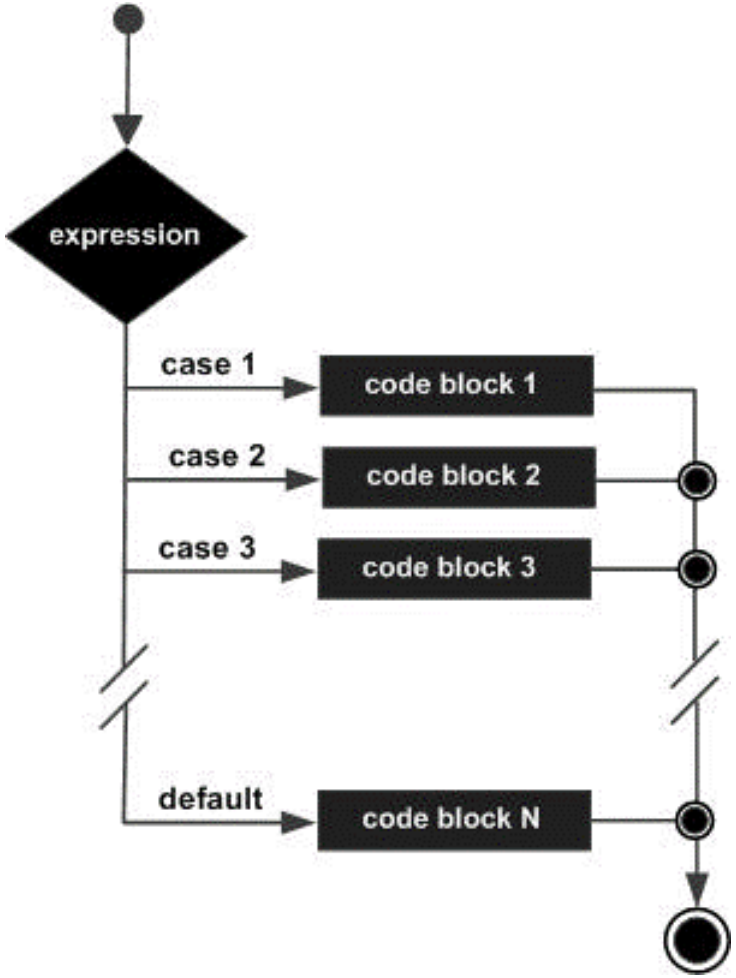
If else if Statement (cont.)

- Example

```
<script>
    var num1 = 10, num2 = 5, num3 = 1;
    if (num1 > num2 && num1 > num3) {
        alert(num1 + " is greatest number");
    }
    else if (num2 > num1 && num2 > num3) {
        alert(num2 + " is greatest number");
    }
    else {
        alert(num3 + " is greatest number");
    }
</script>
```

Switch Statement

- Used to specify **multiple conditional branches**
- Evaluates an **expression** and executes code as a result of a matching case
- An elegant alternative to several if…else...if statements



Switch statement (cont.)

- Syntax

```
switch(expression) {
    case n:
        code block
        break;
    case m:
        code block
        break;
    default:
        code block
}
```

Switch statement (cont.)

```
<script>
    var dayName;
    var day = 5;
    switch (day) {
    case 1: dayName = "Monday";
    break;
    case 2: dayName = "Tuesday";
    break;
    case 3: dayName = "Wednesday";
    break;
    case 4: dayName = "Thursday";
    break;
    case 5: dayName = "Friday";
    break;
    case 6: dayName = "Saturday";
    break;
    case 7: dayName = "Sunday";
    break;
    default:
    dayName = "Invalid day";
    }
    console.log(dayName);
    </script>
```

What are Loops?

Used to execute the same block of code a **specified number of times or while a specified condition is true**

JavaScript supports different kinds of loops:

- **for**
- **for..in**
- **while**
- **do..while**

For Loop

- Can execute a block of code a number of times
- Syntax

```
for (loop init; test; increment/decrement) {
    code block to be executed
}
```

- three important parts:
  - **loop initialization**
  - **test statement**
  - **increment/decrement expression**
- Example

```
<script>
    for (var count = 1; count <= 10; count++) {
        console.log("Current Count : " + count);
    }
</script>
```

For/in Loop

- Used to loop through an object's properties
- Syntax

```
for (var variableName in object){
   statement or block to execute
}
```

- On each iteration, one property from **object** is assigned to **variableName** and loop continues till all the properties of the object are iterated.
- Example

```
<script>
    for (var propertyName in document) {
        console.log(propertyName);
    }
</script>
```

While Loop

- Used to execute a statement or code block repeatedly as long as an **expression** is true
- loop terminates once the expression becomes **false**
- Syntax

```
while (expression){
   Statement(s) to be executed if expression is true
}
```

- Examples

```
<script>
    var count = 1;
    while (count <= 10) {
        console.log("Current Count : " + count);
        count++;
}
</script>
```

Do.while Loop

- Similar to the **while** loop except that the condition check happens at the end of the loop
- Executed at least once even if the condition is false
- Syntax

```
do {
   Statement(s) to be executed;
} while (expression);
```

- Example

```
<script>
    var count = 1;
    do {
        console.log("Current Count : " + count);
        count++;
    } while (count <= 5);
</script>
```

Loop Control Statements

- Used to control the loop execution like starting next iteration early to exiting the loop
- JavaScript provides **break** and **continue** statements

Break Statement

- Used to exit a loop early, breaking out of the enclosing curly braces
- break is also used in switch statement

- Example

```
<script>
    var counter = 1;
    while (counter < 20) {
        if (counter == 6.5) {
            console.log("breaking loop");
            break; // breaks out of loop completely
        }
        counter += .5;
        console.log(counter);
    }
</script>
```

Continue Statement

- breaks one iteration
- Immediately starts the next iteration of the loop and skip the remaining code block
- Example

```
<script>
    for (var i = 20; i < 40; i++) {
        if (i === 30) { continue; }
        console.log(i);
    }
</script>
```

Email us - support@acadgild.com